An hybrid algorithm for data compression

Amir Z. Averbuch[1], Valery A. Zheludev[1], Moshe Guttmann[1], Dan D. Kosloff[2]
[1]School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel
[2]Department of Earth and Planetary Sciences, Tel Aviv University, Tel Aviv 69978, Israel

Abstract

We present an algorithm that compresses two-dimensional data arrays, which are piece-wise smooth in one direction and have oscillating events in the other direction. Seismic, hyper-spectral and fingerprints data have this mixed structure. The transform part of the compression process is an algorithm that combines wavelet and the local cosine transform (LCT) also called an hybrid transform. The quantization and the entropy coding parts of the compression were taken from the SPIHT codec. To efficiently apply the SPIHT codec to a mixed coefficients array, reordering of the LCT coefficients takes place. This algorithm outperforms other algorithms that are based only on the 2D wavelet transforms. Its compression capabilities are also demonstrated on multimedia images that have a fine texture. The wavelet part in the mixed transform of the hybrid algorithm utilizes the library of Butterworth wavelet transforms.

1 Introduction

3D seismic data and hyper-spectral images are considered to be very large datasets. Efficient algorithms are needed to compress these datasets in order to store or transmit them from planes, boats and satellites to base stations for further analysis. The compression ratio should be as high as possible without damaging the interpretations procedures that takes place after decompression. The compression of these datasets should preserve fine details. Since wavelet transforms have a successful record in achieving high compression ratios for still images, then these techniques were ported to handle seismic compression [14, 19, 27]. However, the outcomes were less impressive due to the great variability of seismic data, the inherent noisy background and their oscillatory nature. Moreover, some researchers argued that seismic signals are not wavelet-friendly [1, 20]. This happens because of the oscillatory patterns that are present in seismic data.

The local cosine transform (LCT) [1, 20, 28], which uses the lapped DCT-IV transform [13] with an adaptive partition, was applied to compress 2D seismic sections. The LCT catches well oscillatory patterns. A DCT-II–based algorithm that compresses segmented seismic cube was presented in [8].

Seismic data has different structure in its vertical and horizontal directions. While horizontal structure is piece-wise smooth, the vertical traces comprise oscillatory patterns. Therefore, wavelets provide a sparse representation of seismic data in the horizontal direction but fail to properly represent vertical oscillations. Since LCT handles much better oscillatory patterns, we propose to apply the wavelet transform to the horizontal direction and the LCT to the vertical direction.

A typical data compression scheme contains three phases: 1. Application of a transform to the image (lossless phase). 2. Quantization (lossy phase). 3. Entropy coding (lossless phase).

EZW [25] and SPIHT [22] are well known wavelet based schemes for 2D compression. Both utilize the correlation among the multiscale decomposition of an image to achieve high compression ratio. SPIHT scheme is associated with wavelet transforms that rely on the space-frequency localization of the wavelets and the tree structure of the coefficients arrays in its multiscale representation. In this paper, we propose to use the SPIHT coding for the last two phases in a general compression scheme: quantization and entropy coding. The way the transform is used and applied is new. The new transform produces mixed LCT and wavelet coefficients that are submitted to the SPIHT coding for quantization and entropy coding. The LCT coefficients get spatial meaning by partition the data in the vertical direction. The joint coefficients array are organized in a wavelet tree-like way by reordering the LCT coefficients. Therefore, this scheme is called an *hybrid compression algorithm*. If there are oscillations in both directions, then the LCT is applied to both directions followed by reordering of the transform coefficients and then SPIHT (quantization and entropy coding) is applied. There are cases, when multiscale LCT transform is applied (see [1]).

The proposed algorithm proved to be efficient also for compressing hyper-spectral data cubes. An hyper-spectral data cube can be captured by a special camera installed in a flying device such as an airplane or a satellite. The hyper-spectral camera captures images of the ground surface in many ($\approx$ 200) wavebands. Thus, each spatial pixel is represented by a vector (also called multipixel) of the intensities in all the available wavebands. Compression of hyper-spectral cubes should retain the spectral characteristic features of the multipixels. The sizes of these data cubes types are huge and high quality high compression ratio is needed. A number of compression algorithms were suggested ([12]). Many of them extend wavelet-based compression methods to compress 3D hyper-spectral data cubes [7, 11, 9]. However, an hyper-spectral camera collects each time a line of about $\approx$300 multipixels. Thus, we argue that, for the transmission the data from the receiver to the processing center, the 2D compression of the planes $wavebands \times multipixels$ is preferable.

We get that the performance of the hybrid algorithm outperforms non-hybrid algorithms such as wavelet–based algorithms.

Another successful application of the hybrid algorithm is compression of fingerprints images. The FBI uses the compression standard [6] that is based on the 9/7 biorthogonal 2D wavelet transform

[10]. This transform, which uses finite impulse response (FIR) filters of length 7 and 9, is included also in JPEG 2000 image compression standard [18]. The hybrid algorithm produces higher PSNR results and retains better the structure of the fingerprints compared to the outputs from the application of 2D wavelet transforms.

For multimedia type images, which are relatively smooth, the hybrid performance was close (sometimes even inferior) to the performance of the 2D wavelet- based algorithms. On the other hand, the hybrid algorithm outperforms the 2D wavelet based algorithms on images that have fine texture such as "Barbara". The hybrid algorithm retains the texture even in a very low bit rate.

The 9/7 biorthogonal wavelet transform is widely used in image processing applications. However, our experiments demonstrate that in many cases the biorthogonal wavelet transforms, which are based on the infinite impulse response (IIR) Butterworth filters [2, 3], provide better compressed images than what the 9/7 transforms based algorithms produce. This is true for the 2D wavelet transforms as well as for the hybrid transforms. We compare among the performances of different 2D wavelet transforms and compare also among 2D wavelet transforms and the hybrid transforms.

The paper is organized as follows. In Section 2, we recall some known facts about LCT, wavelet transforms and SPIHT coding. Section 3 describes the hybrid algorithm. Section 4 presents experimental results on compression of seismic sections, hyper-spectral data cubes, fingerprints and multimedia images. An outline of the Butterworth wavelet transforms is given in the Appendix.

2 Preliminaries

2.1 Local cosine transform (LCT)

The following two DCT types are used in image compression:

1. DCT-II, which is used in JPEG, of a signal $\mathbf{f} = \{f_n\}_{n=0}^{N-1},\ N = 2^p$, is:

$$\hat{f}^{II}(k) = b(k) \sum_{n=0}^{N-1} f_n \cos\left[\frac{k\pi}{N}\left(n+\frac{1}{2}\right)\right], \quad b(k) = \begin{cases} 1/\sqrt{2}, & \text{if } k=0; \\ 1, & \text{otherwise,} \end{cases}$$

$$f_n^{II} = \frac{2}{N} \sum_{k=0}^{N-1} b(k)\hat{f}^{II}(k) \cos\left[\frac{k\pi}{N}\left(n+\frac{1}{2}\right)\right].$$

2. DCT-IV of the signal $\mathbf{f}$ is:

$$\hat{f}^{IV}(k) = \sum_{n=0}^{N-1} f_n \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)\right],$$

$$f_n^{IV} = \frac{2}{N} \sum_{k=0}^{N-1} \hat{f}^{IV}(k) \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)\right].$$

Unlike DCT-II, no coefficient of DCT-IV is the average of the function f. Thus, application of DCT-IV to a flat constant function will result in many non-zero transform coefficients. This is of course a disadvantage of DCT-IV when it is used in transform coding of smooth signals. On the other hand, the DCT-IV is a good choice for coding oscillatory signals.

An important difference between DCT-II and DCT-IV lies in the symmetry of the transform basis functions at the boundaries of their intervals. The cosine element in the basis function of the 1-D DCT-II has the term $\left[\frac{k\pi}{N}\left(n+\frac{1}{2}\right)\right]$, while the cosine element in the 1-D DCT-IV has the term $\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)\right]$. Thus, the difference between them is the addition of $\frac{1}{2}$ to the term k. This difference totally changes the symmetry behavior of the basis functions at the boundaries. By expanding the range of n from $0,\ldots,N-1$ to $-N,\ldots,2N-1$, we get that the basis functions of DCT-II are even on both sides, e.g., even with respect to $-\frac{1}{2}$ and $N-\frac{1}{2}$. Therefore, application of this transform to partitioned signals and images reduces the blocking effect. This was one of the reasons why the 8×8 2D DCT-II transform is used in JPEG image compression standard.

By doing the same to DCT-IV, we get that the DCT-IV basis functions are even on the left side with respect to $-\frac{1}{2}$ and odd on the right side with respect to $N-\frac{1}{2}$. Therefore, direct application of the DCT-IV transforms to a partitioned data leads to severe boundary discrepancies. However, this transform serves as a base for the so called local cosine bases [13], which are the windowed lapped DCT-IV transforms. These bases were successfully exploited in [1, 4, 20, 24, 28] for image compression in general and seismic data compression in particular.

Assume we have a signal $\mathbf{S}\stackrel{\Delta}{=}\{s_k\}_{k=0}^{N-1}$ and some partition P of the interval $0:N-1$. The idea behind the lapped DCT-IV transform also called the local cosine transform (LCT) of a signal, is to apply overlapped bells to adjacent sub-intervals. Then, the overlapping parts are folded back to the sub-intervals across the endpoints of the sub-intervals and the DCT-IV transform on each sub-interval is implemented. In the reconstruction phase, the transform coefficients are unfolded. For details, see [1, 4, 23]. We call this transform the P-based lapped DCT. We illustrate the partition of the line by bells in Fig. 2.1. The choice of a bell is discussed in [4, 13, 24]. The bell we chose is given in section 4.

Figure 2.1: Line partitioning by bells

2.2 Wavelet transforms

Currently, wavelet transform is a recognized tool for image processing applications. In particular, they have gained a proven success in image compression applications. We summarize here some well known facts that are needed later.

The multiscale wavelet transform of a signal is implemented via iterated multirate filtering. One step in the transform of a signal $\mathbf{S} \stackrel{\Delta}{=} \{s_k\}_{k=0}^{N-1}$ of length N consists of filtering the signal by a half-band low-pass filter L and a half-band high-pass filter H, which is followed by factor 2 downsampling of both filtered signals. Thus, two blocks of coefficients $\mathbf{w}_L^1 \stackrel{\Delta}{=} \left\{l_k^1\right\}_{k=0}^{N/2-1}$ and $\mathbf{w}_H^1 \stackrel{\Delta}{=} \left\{h_k^1\right\}_{k=0}^{N/2-1}$ of length $N/2$ each are produced. The coefficients $\mathbf{w}_L^1$ contain the whole information of the low frequency component of the signal $\mathbf{S}$. The coefficients $\mathbf{w}_H^1$ do the same for the high frequency component of the signal $\mathbf{S}$. The blocks $\mathbf{w}_L^1$ and $\mathbf{w}_H^1$ halve the Nyquist frequency band $\mathbf{F}$ of the signal $\mathbf{S}$: $\mathbf{F} \rightarrow \mathbf{F}_L^1 \bigcup \mathbf{F}_H^1$.

On the other hand, the wavelet coefficients have a spatial meaning. The coefficient l_m^1 is the result of weighted averaging of the set $\mathbf{S}_m^{1,\lambda}$ of signal's samples. The set $\mathbf{S}_m^{1,\lambda}$ is centered around the sample s_{2m} of the signal $\mathbf{S}$ and its scope λ^1 is equal to the width of the impulse response (IR) of the filter L, provided L is a FIR filter. If L is a IIR filter whose IR decays rapidly, then λ equals to the effective width of the IR of the filter L. The coefficient h_m^1 is the result of numerical differentiation of some order d of the signal $\mathbf{S}$ at the point $2m+1$. The coefficient h_m^1 is, in a sense, the result of numerical differentiation of some order d of the signal $\mathbf{S}$ at the point $2m+1$. For this, the set $\mathbf{S}_m^{1,\chi}$ of signal's samples is involved, whose scope χ^1 is equal to the (effective) width of the IR of the filter H.

The next step of the wavelet transform applies the pair of filters L and H to the coefficients array $\mathbf{w}_L^1$. The filtering is followed by downsampling. The produced blocks of coefficients $\mathbf{w}_L^2 \stackrel{\Delta}{=} \left\{l_k^2\right\}$ and $\mathbf{w}_H^2 \stackrel{\Delta}{=} \left\{h_k^2\right\}$ from L and H, respectively, are of length $N/4$ each, halve the sub-band $\mathbf{F}_L^1 \longrightarrow \mathbf{F}_L^2 \bigcup \mathbf{F}_H^2 \Longrightarrow \mathbf{F}_L \longrightarrow \mathbf{F}_L^2 \bigcup \mathbf{F}_H^2 \bigcup \mathbf{F}_H^1$. In the time domain, the coefficient l_m^2 is the result from averaging the set of samples $\mathbf{S}_m^{2,\lambda}$. The set $\mathbf{S}_m^{2,\lambda}$ is centered around the sample s_{4m} of the signal $\mathbf{S}$ and its scope is $\lambda^2 \approx 2\lambda^1$. Similarly, the coefficient h_m^2 is associated with the set $\mathbf{S}_m^{2,\chi}$ of samples, which is is centered around the sample s_{4m+2} of the signal $\mathbf{S}$ and its scope is $\chi^2 \approx 2\chi^1$. Note that the set $\mathbf{S}_m^{2,\chi}$ occupies approximately the same area as the pair of its "offspring" sets $\mathbf{S}_{2m}^{1,\chi}$ and $\mathbf{S}_{2m+1}^{1,\chi}$.

Then, this decomposition is iterated to reach scale J. It produces the coefficients array $\mathbf{w}^J$ whose structure is

$$\mathbf{w}^J = \mathbf{w}_L^J \bigcup \mathbf{w}_H^J \bigcup \mathbf{w}_H^{J-1} \bigcup \ldots \bigcup \mathbf{w}_H^1. \tag{2.1}$$

Respectively, the Nyquist frequency band $\mathbf{F}$ is split into subbands whose widths are distributed in a logarithmic way to become

$$\mathbf{F} \longrightarrow \mathbf{F}_L^J \bigcup \mathbf{F}_H^J \bigcup \mathbf{F}_H^{J-1} \bigcup \ldots \bigcup \mathbf{F}_H^1. \tag{2.2}$$

The diagram of a three-scale wavelet transform and the layout of the transform coefficients are dis-

played in Fig. 2.2.

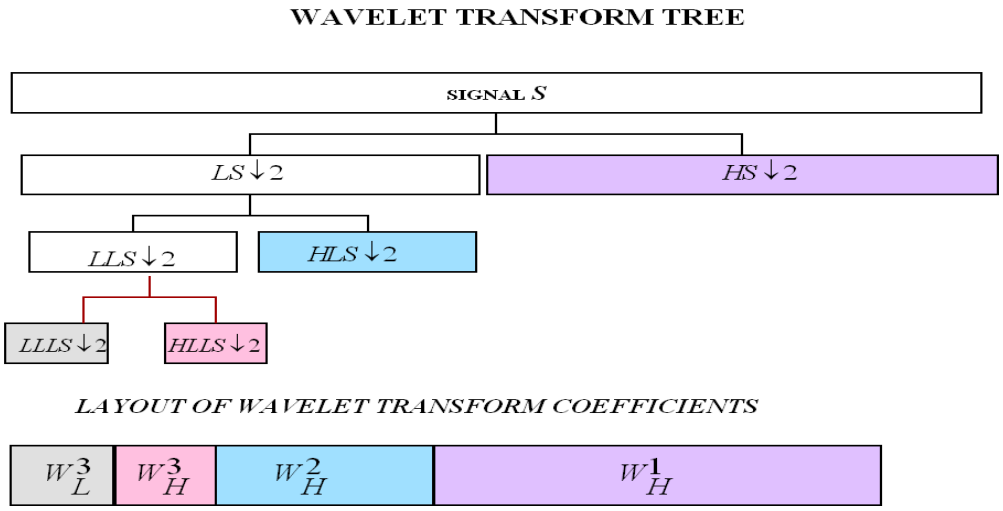

Figure 2.2: Three-scale wavelet transform and the layout of the transform coefficients

The wavelet transform of a two-dimensional array $\mathbf{T} = \{t_{n,m}\}$ of size $N \times M$ is implemented in a tensor product way. First, the pair of filters L and H is applied to the columns of $\mathbf{T}$ and the results are downsampled. The coefficients arrays $\mathbf{L}$ and $\mathbf{H}$ of size $N/2 \times M$ are produced. Then, the filters L and H are applied to the rows of $\mathbf{L}$ and $\mathbf{H}$. This filtering is followed by downsampling which results in four sub-arrays coefficients **LL**, **LH**, **HL** and **HH** of size $N/2 \times M/2$. The 2D Nyquist frequency domain is split accordingly. Then, the above procedure is applied to the coefficient array **LL** to produce the subarrays **(LL)LL**, **(LL)LH**, **(LL)HL** and **(LL)HH** of size $N/4 \times M/4$. Then, this procedure is iterated using **(LL)LL** instead of **LL** and so on. The layout of the transform coefficients, which corresponds to the Nyquist frequency partition, for the three-scale wavelet transform, is displayed in Fig. 2.3.

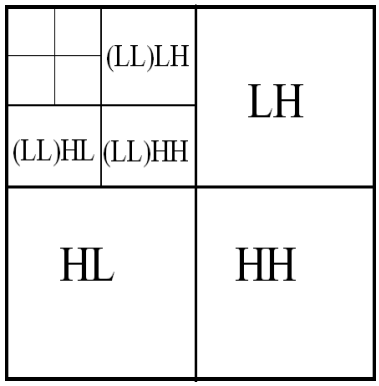

Figure 2.3: Layout of the coefficients of the three-scale of a 2D wavelet transform

2.3 SPIHT coding

The coefficients from **(LL)LH**, **(LL)HL** and **(LL)HH** are produced by the same operations as the coefficients from **LH**, **HL** and **HH**, respectively. The coefficient $c_{nm}^{llhl} \in$ **(LL)HL** from the second scale is associated with the set $\mathbf{P}_{nm}^{llhl}$ of pixels that is centered around the pixel $p_{4n+2,4m}$. The set $\mathbf{P}_{nm}^{llhl}$ occupies, approximately, the same area that the four sets $\mathbf{P}_{\nu,\mu}^{hl}$, which are associated with the first scale coefficients $c_{\nu,\mu}^{hl} \in$ **HL**, where $\nu = 2n+1, 2n+3$, $\mu = 2m-2, 2m+2$, occupy. In this sense, $c_{\nu,\mu}^{hl} \in$ **HL** are the descendants of the coefficient $c_{nm}^{llhl} \in$ **(LL)HL**. Similar relations exist between the coefficients from **LH**, **HH** and **(LL)LH**, **(LL)HH** and also between the coefficients from other adjacent scales. These relations are illustrated in Fig. 2.4.

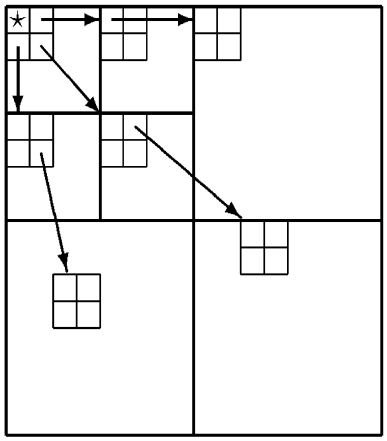

Figure 2.4: Relationship among wavelet coefficients from different scales

This ancestor-descendant relationship between wavelet coefficients in different scales, which are located at the same spatial area, is exploited in the embedded zerotree wavelet (EZW) codec [25]. This codec takes advantage of the self-similarity between wavelet coefficients across the decomposed scales and their decay toward high frequency scales. The EZW codec relates the coefficients with quad-trees. One of most efficient algorithms, which is based on the zerotree concept, is the SPIHT algorithm [22]. They added to the EZW algorithm a set partitioning technique. This algorithm combines adaptive quantization of the wavelet coefficients with coding. The algorithm presents a scheme for progressive coding of coefficient values when the most significant bits are coded first. This property allows to control the compression rate. The produced bitstream is further compressed losslessly by the application of adaptive arithmetic coding. The SPIHT coding procedure is fast and its decoding procedure is even faster.

3 The hybrid algorithm

We propose to apply the wavelet and the LCT transforms in the horizontal and vertical directions, respectively. The mixed wavelet and LCT transforms coefficients are encoded by the SPIHT algorithm. For this, we have to establish an ancestor-descendant relationship between the transform coefficients that will be similar to the relationships that exist between the 2D multiscale wavelet transform coefficients. We reorder the coefficients of the LCT in a way that mimics the layout of the 2D wavelet transform coefficients. The array of 2^k LCT coefficients are separated into $k+1$ blocks according to a logarithmic scale in the following way:

$$\mathbf{c} \triangleq \Big(\; c_0 \mid \quad c_1 \mid \quad c_2 \; c_3 \mid \quad c_4 \; c_5 \; c_6 \; c_7 \mid \quad c_8 \; c_9 \; c_{10} \; c_{11} \; c_{12} \; c_{13} \; c_{14} \; c_{15} \; \Big)^T. \tag{3.1}$$

The partition in Eq. (3.1) appears automatically when the coefficients indices are presented in a binary mode:

$$\mathbf{c} = \Big(\; c_0 | \quad c_1 | \quad c_{10} \, c_{11} | \quad c_{100} \, c_{101} \, c_{110} \, c_{111} | \quad c_{1000} \, c_{1001} \, c_{1010} \, c_{1011} \, c_{1100} \, c_{1101} \, c_{1110} \, c_{1111} \; \Big)^T. \tag{3.2}$$

Thus, the array is partitioned according to the number of bits in the coefficients indices. We call this a bit-wise partition.

Assume we are given $N \times M$ data array $\mathbf{T}$, where $N = 2^k Q$, $M = 2^J R$. We define the partition P of the interval $I \triangleq [0, 1, ..., N-1]$ by splitting it into Q subintervals $I = \bigcup_{i=1}^{Q} I^i$ of length 2^k each. We apply the P-based LCT transform to each column of $\mathbf{T}$. Thus, the array $\mathbf{T}$ is transformed into the array $\mathbf{C}$ of LCT coefficients.

For each column, we get the array $\mathbf{c}$ of N LCT coefficients, which consists of Q blocks $\mathbf{c} = \bigcup_{i=1}^{Q} \mathbf{c}^i$, where $\mathbf{c}^i \triangleq \{c_n^i\}_{n=0}^{2^k-1}$. Each of them can be bit-wise partitioned as in Eq. (3.2): $\mathbf{c}^i = \bigcup_{\beta=0}^{k} \mathbf{b}_\beta^i$, where $\mathbf{b}_0^i = c_0^i$, $\mathbf{b}_1^i = c_1^i$ and $\mathbf{b}_\beta^i$ is the set of coefficients c_n^i, whose indices can be represented by β bits.

In order to obtain a wavelet-like structure of the array $\mathbf{c}$ of N LCT coefficients, we rearrange it in a bit-wise mode

$$\mathbf{c} \longrightarrow \mathbf{b} \triangleq \bigcup_{\beta=0}^{k} \mathbf{b}^\beta \quad \mathbf{b}^\beta \triangleq \bigcup_{i=1}^{Q} \mathbf{b}_i^\beta. \tag{3.3}$$

Thus, we get $\mathbf{b}^0 \triangleq \left(c_0^1, c_0^2, ..., c_0^Q\right)^T$, $\mathbf{b}^1 \triangleq \left(c_1^1, c_1^2, ..., c_1^Q\right)^T$, $\mathbf{b}^2 \triangleq \left(c_2^1, c_3^1; c_2^2, c_3^2; ...; c_2^Q, c_3^Q\right)^T$, and so on. This rearrangement is illustrated in Fig. 3.1. The structure of the array $\mathbf{b}$ is similar to the structure of the coefficients array $\mathbf{w}$ of the wavelet transform, where the transform was decomposed into the $k-1$ scale. The similarity relations are

$$\mathbf{b}^0 \longleftrightarrow \mathbf{w}_L^{k-1}, \; \mathbf{b}^1 \longleftrightarrow \mathbf{w}_H^{k-1}, \; \mathbf{b}^2 \longleftrightarrow \mathbf{w}_H^{k-2}, \; ..., \; \mathbf{b}^{k-1} \longleftrightarrow \mathbf{w}_H^1. \tag{3.4}$$

The ancestor-descendant relationship in the array $\mathbf{b}$ are similar to that in $\mathbf{w}$.

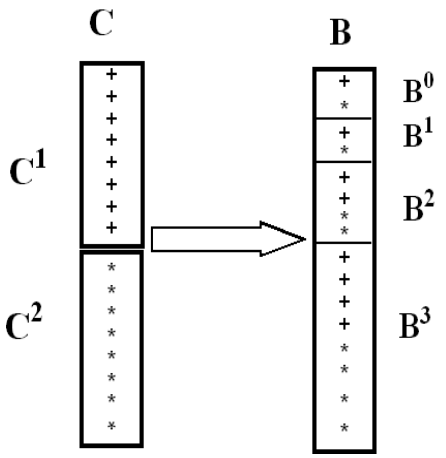

Figure 3.1: Reordering scheme of the LCT coefficients

We perform this reordering of the LCT coefficients for all the columns of the array **C**. Thus, we get

$$\mathbf{C} \longrightarrow \mathbf{B} = \bigcup_{\beta=0}^{k} \mathbf{B}^{\beta} \quad \mathbf{B}^{\beta} \triangleq \bigcup_{i=1}^{Q} \mathbf{B}_i^{\beta}. \tag{3.5}$$

Then, each row of the array **B** is decomposed till scale J by the application of the wavelet transform. This produces the hybrid LCT–wavelet coefficients array denoted as **CW**. The structure of **CW** is similar to the the structure of a 2D wavelet coefficients array, where the transform on columns was decomposed to scale $k-1$, while the transform on the rows was decomposed till scale J. Three scales decomposition structure of the **CW** array is illustrated in Fig. 3.2.

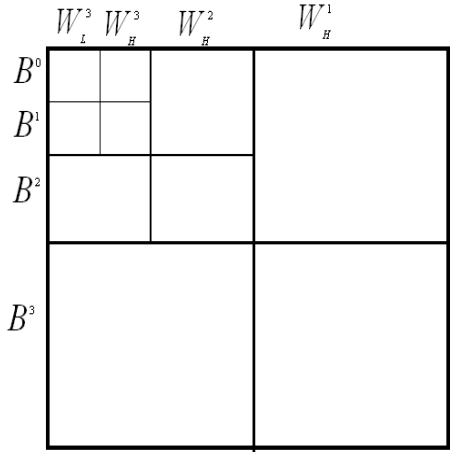

Figure 3.2: Layout of the LCT–wavelet coefficients in three scales

The array **CW** is the input to SPIHT that uses quantization and entropy coding only.

If the data arrays have oscillating structures in both vertical and horizontal directions, then, we apply the LCT transform in both directions followed by reordering of the transform coefficients in both directions as was explained above.

4 Experimental results

The hybrid algorithm was applied to compress different data types. It produced very good results for seismic and hyper-spectral images. The hybrid algorithm outperforms compression algorithms that are based on the application of the 2D wavelet transform followed by either SPIHT or EZW codecs where the bit rate was the same in the experiments. Seismic events in traces were retained much better. It is true for raw gathers as well as for stacked seismic sections.

In addition, it was applied to a fingerprint image and multimedia images such as "Barbara" that contains fine texture. Unlike the 2D wavelet- based algorithm, the hybrid algorithm retains the texture even in a very low bit rate.

The choice of a bell has some effect on the performance of the LCT and, consequently, on the hybrid algorithm. A library of bells was introduced in [24]. A comparative study of their effects on the performance of an LCT–based image compression algorithm was given in [21]. However, such a comparison is beyond the scope of this paper and we did not check the effects of different bells. Our goal was to demonstrate the capabilities of the new method. For this purpose it was sufficient to implement LCT with the "sine" bell $b(x) = \sin \frac{\pi}{2}(x + 1/2)$.

Transforms notation In our experiments we use the following transforms:

W9/7 – The 2D 9/7 wavelet transform.

WButt/M – The 2D Butterworth wavelet transform with M vanishing moments.

H9/7/Q – The Hybrid transform with the 1D 9/7 wavelet transform in the horizontal direction and the LCT in the vertical direction, where the partition of the section was into Q horizontal rectangles. The transform coefficients were reordered as explained in section 3 to fit SPIHT encoding.

HButt/M/Q – The Hybrid transform with the 1D Butterworth wavelet transform with M vanishing moments in the horizontal direction and the LCT in the vertical direction. The LCT uses partition of the vertical direction into Q horizontal rectangles. The transform coefficients were reordered as was explained in section 3 to fit SPIHT encoding.

LDCT/Q/R – The 2D LCT transform, which uses partition of the image into $Q \times R$ rectangles. The transform coefficients were reordered as was explained in section 3 to fit SPIHT encoding.

The peak signal to noise ratio (PSNR), which is used to evaluate the quality of the reconstructed image, is

$$PSNR = 10 \log_{10} \left(\frac{N\, M^2}{\sum_{k=1}^{N} (x_k - \tilde{x}_k)^2} \right) \; dB, \tag{4.1}$$

where N is the total number of samples, x_k is an original sample and $\tilde{x}_k$ is the reconstructed sample, $M = \max_{k=1,\ldots,N} \{|x_k|\}$. The performance of the 2D wavelet transforms was compared with the performance of the hybrid transforms using different wavelets. The SPIHT codec we utilized was enhanced with an adaptive arithmetic coder. The bolded numbers in the tables are the best achievable results.

4.1 Seismic sections compression

Compression of seismic data, which is used for reconstruction of subsurface layers structure, should retain all the seismic events in the traces.

4.1.1 Stacked CMP data section

We used a stacked common mid point (CMP) data section in the experiments. Each pixel has 32 bits. We display the original section of size 512×512 and a fragment of size 200×200 in Fig. 4.1.

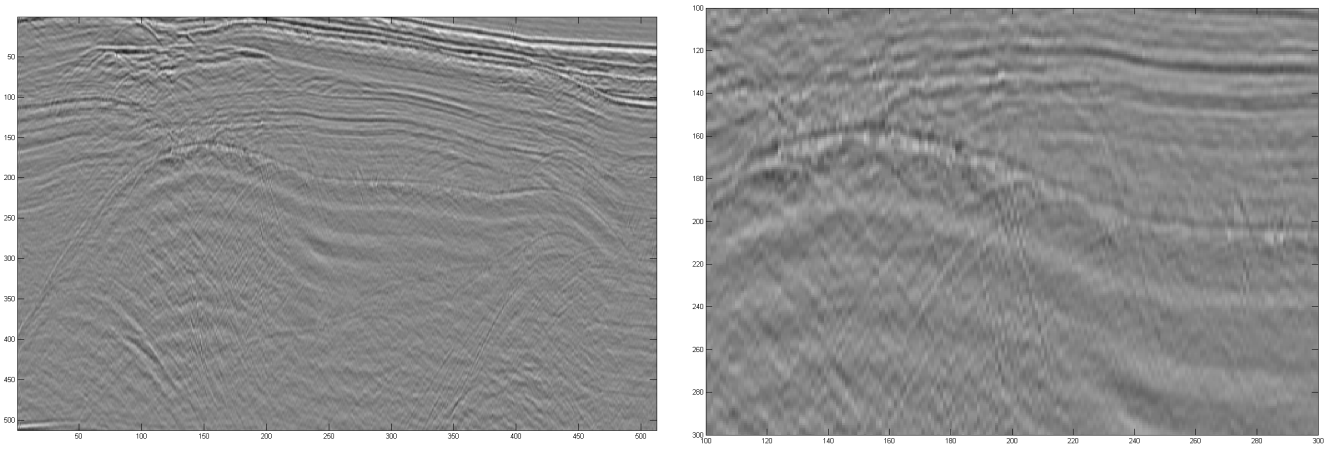

Figure 4.1: The original stacked CMP section. Left: the whole section. Right: a fragment from the left image

Five types of transforms were used: **W9/7**, **WButt/4**, **WButt/10**, **H9/7/8** and **HButt/10/8**. The achieved PSNR values are given in Table 4.1.

bit/pixel (compression ratio)	W9/7	WButt/4	WButt/10	H9/7/8	HButt/10/8
1/4 (128)	27.19	27.09	27.43	28.01	**28.06**
1/2 (64)	29.82	29.82	30.29	31.27	**31.77**
1 (32)	33.80	34.01	34.51	37.03	**37.19**
2 (16)	42.00	43.02	44.89	45.84	**46.38**

Table 4.1: PSNR values after decompression of the stacked CMP seismic section. Each pixel has 32 bits.

At low bitrate, the performance of the **WButt/4** wavelet transform with 4 vanishing moments is close to the performance of **W9/7** transform, which also has 4 vanishing moments. Figure 4.2 displays fragments of the reconstructed stacked CMP section after the 2D **W9/7** and **HButt/10/8** transforms were applied and the coefficients were compressed by SPIHT with 1/2 bit pet pixel. The fragment from the **HButt/10/8** transformed section retains the structure of the data unlike the **W9/7** based fragment.

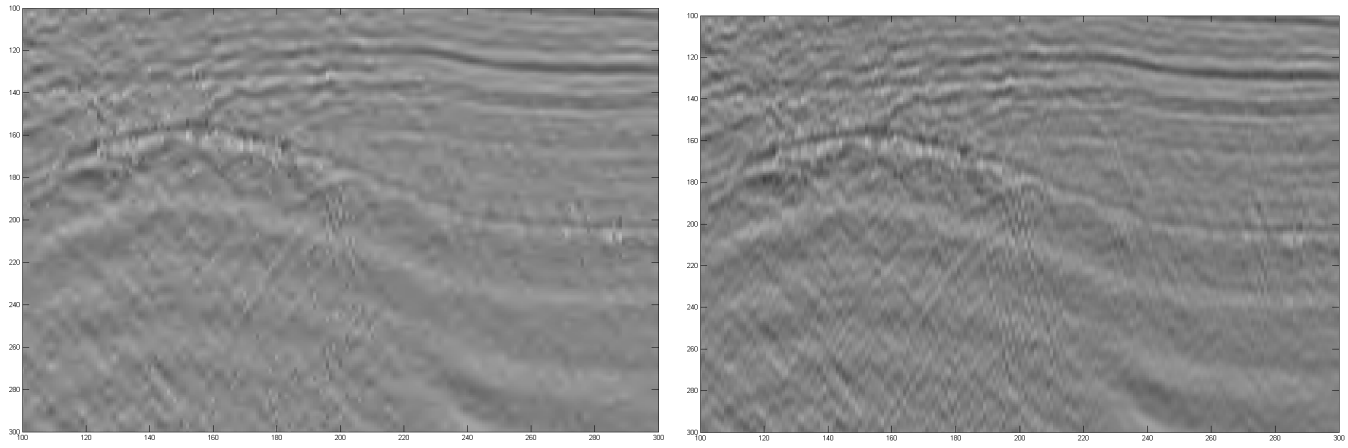

Figure 4.2: A fragment of the reconstructed CMP section. Left: The 2D 9/7 wavelet transform **W9/7** was applied. Right: The Hybrid transform **HButt/10/8** was applied. The transforms coefficients were encoded by SPIHT assuming 1/2 bit per pixel compression rate.

Figure 4.3 displays a fragment of trace #200 from the original section versus the fragment from the section reconstructed after the application of the **W9/7** transform. Figure 4.4 does the same using the fragment from the section reconstructed after the application of the **HButt/10/8** transform.

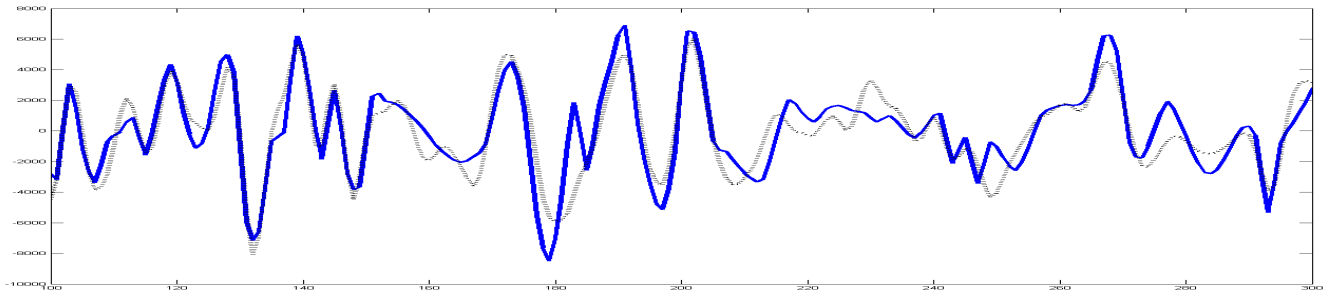

Figure 4.3: A fragment from trace #200 from the CMP section. Dotted line: Original. Solid line: Restored after the application of the **W9/7** transform and SPIHT encoding assuming 1/2 bit per pixel compression rate

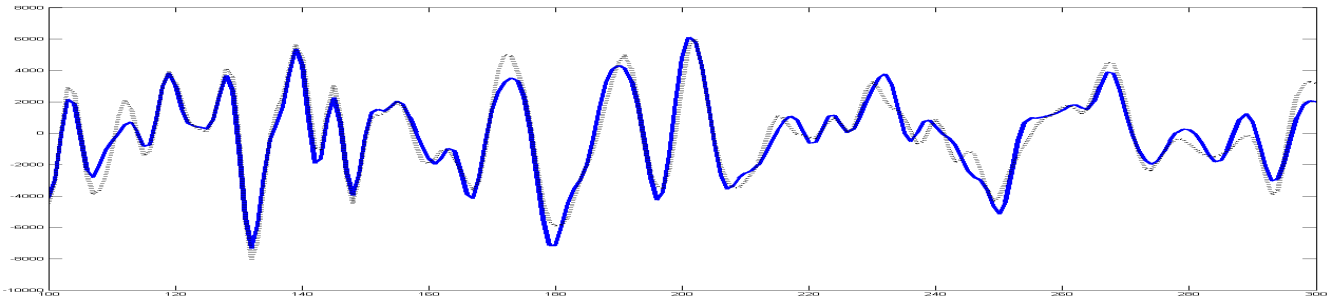

Figure 4.4: A fragment from the trace #200 from the CMP section. Dotted line: Original. Solid line: Restored after **HButt/10/8** transform and SPIHT encoding using 1/2 bit per pixel compression rate

We observe that the reconstructed trace in Fig. 4.4 is very close to the original trace. This is not the case for the trace in Fig. 4.3.

4.1.2 Marine shot gather data section

In another seismic experiments, we used a marine shot gather (MSG) data section of size 640×512. The PSNR of this data is lower than the stacked section even when the subsurface layers are not that distinct. As before, we compared between the performance of the 2D wavelet transforms and the hybrid transforms that use different wavelets. Each data pixel has 32 bits. We display the original section of size 640×512 and fragment of size 300×300 from it in Fig. 4.5.

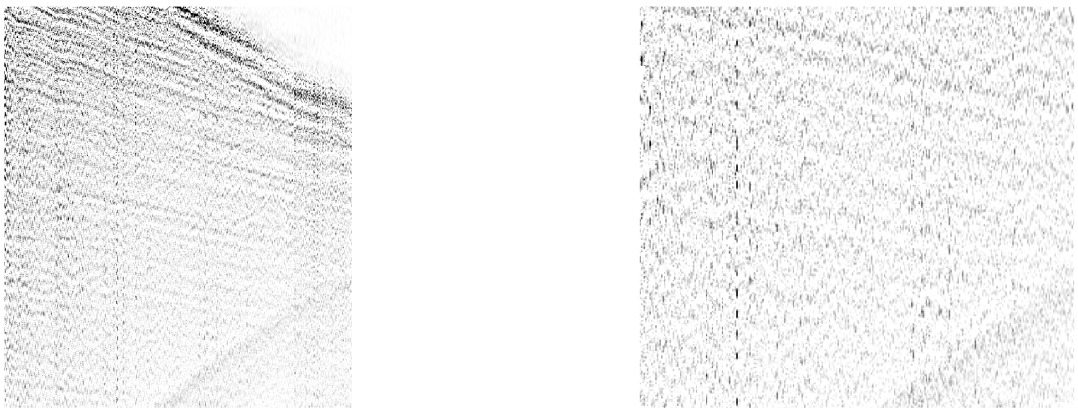

Figure 4.5: The original marine shot gather (MSG) section. Left: The whole section. Right: A fragment from the section on the left

As in section 4.1.1, we compare between the performance of five transforms. The only difference was that the Hybrid transforms partition into $Q = 10$ horizontal rectangles of height 64 each instead of $Q = 8$. The achieved PSNR values are presented in Table 4.2.

bit/pixel (compression ratio)	W9/7	WButt/4	WButt/10	H9/7/10	HButt/10/10
1/4 (128)	26.18	25.96	26.12	26.51	**26.59**
1/2 (64)	27.99	27.76	27.90	28.78	**29.04**
1 (32)	31.63	31.32	31.57	32.52	**32.76**
2 (16)	37.71	37.42	37.77	39.91	**40.04**

Table 4.2: PSNR values for compression of the MSG seismic section. Each pixel has 32 bits

Figure 4.6 displays the fragments of the reconstructed MSG section after the application of 2D **W9/7** and Hybrid **HButt/10/10** transforms when the SPIHT encoded the coefficients into 1 bit per pixel compression ratio.

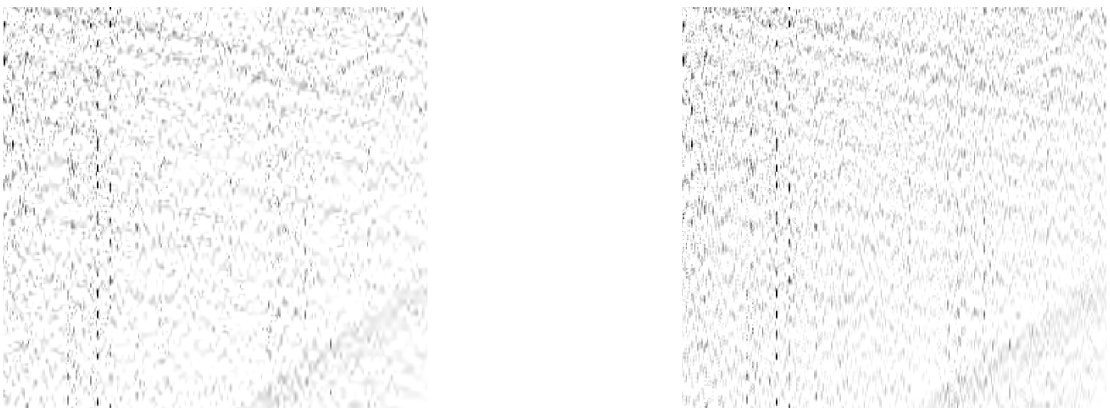

Figure 4.6: A fragment from the reconstructed MSG section. Left: Reconstruction from the application of the **W9/7** transform. Right: The reconstruction from the application of the Hybrid **HButt/10/10** transform. The transforms coefficients were encoded/decoded by SPIHT using 1 bit per pixel compression rate

Figure 4.7 displays a fragment of trace #300 from the original section versus the fragment from the section reconstructed after the application of **W9/7** transform. Figure 4.8 does the same by using the fragment from the section reconstructed after the application of **HButt/10/10** transform.

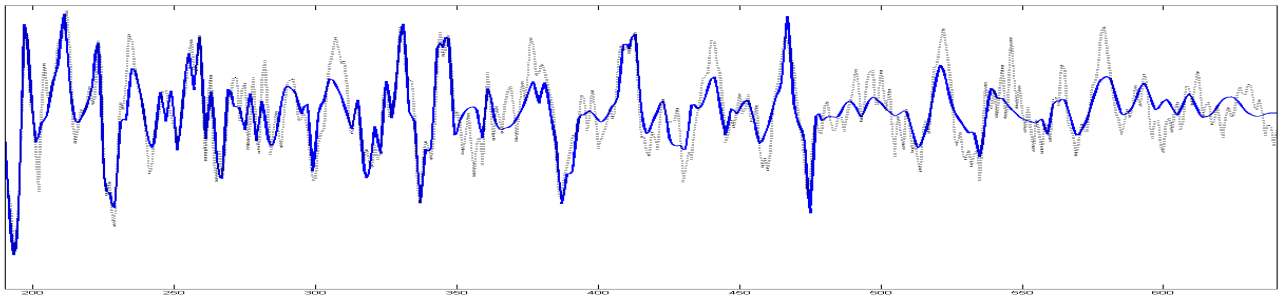

Figure 4.7: A fragment from trace #300 of the MSG section. Dotted line: Original. Solid line: Restored after the application of the **W9/7** transform with SPIHT encoding/decoding using 1 bit per pixel compression ratio

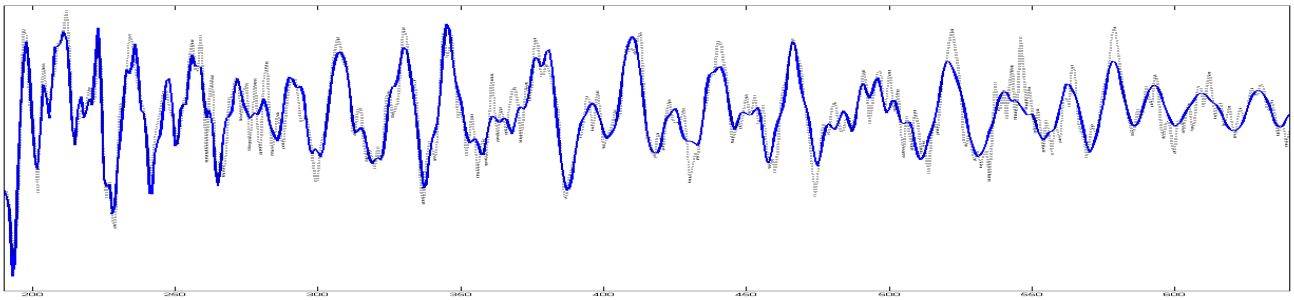

Figure 4.8: A fragment from trace #300 of the MSG section. Dotted line: Original. Solid line: Restored after the application of the **HButt/10/10** transform and SPIHT encoding/decoding using 1 bit per pixel compression ratio

We see that the reconstructed trace in Fig. 4.8 is much closer to the original trace in comparison to the trace in Fig. 4.7. The accuracy is better for 2 bits per pixel. The **HButt/10/10** transform (Fig. 4.10) restored all the seismic events more accurately in comparison to **W9/7** transform (Fig. 4.9).

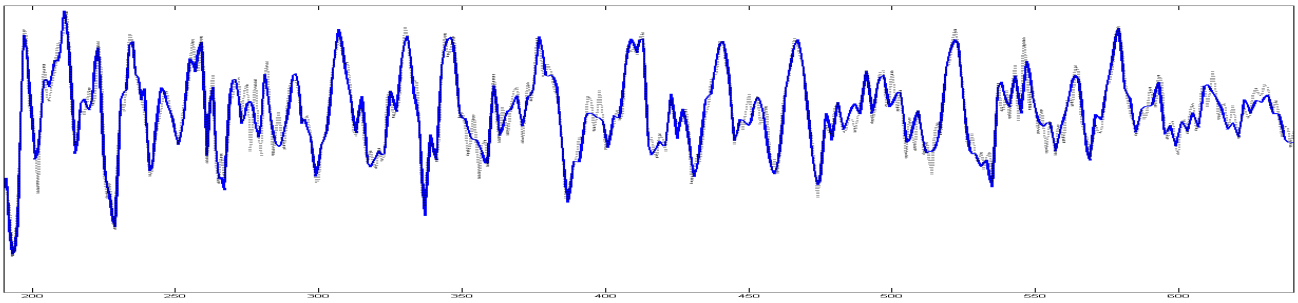

Figure 4.9: A fragment from trace #300 of the marine shot gather section. Dotted line: Original. Solid line: Restored after the application of the **W9/7** transform and SPIHT encoding/decoding using 2 bits per pixel compression ratio

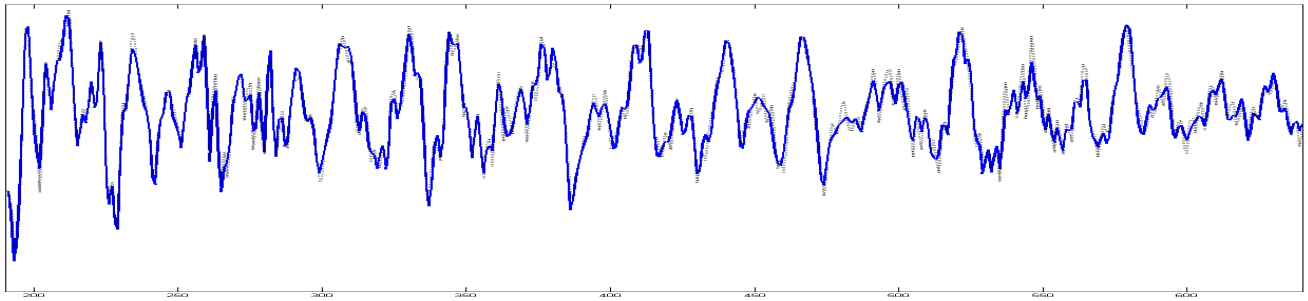

Figure 4.10: A fragment from trace #300 of the marine shot gather section. Dotted line: Original. Solid line: Restored after the application of the **HButt/10/10** transform and SPIHT encoding/decoding using 2 bits per pixel compression ratio

Conclusion: For seismic data, the Hybrid compression algorithm significantly outperforms compression algorithms that are based on the 2D wavelet transforms.

4.2 Compression of hyper-spectral data

The hybrid compression was applied to hyper-spectral data cubes. These cubes were captured from a plane that took simultaneously pictures in many (≈ 200) wavebands from the ground surface. This type of camera can be placed also in a satellite. Thus, a spectrum of intensities for all the wavebands is assigned to each spatial pixel, which is called multipixel. When hyper-spectral data is compressed, it is important to preserve the spectral characteristic features of the multipixels. An hyper-spectral image is treated as a 3D cube where X,Y are the spatial axes and Z is the waveband axis. We compress the 2D $X-Z$ plane by applying the hybrid transform followed by the application of SPIHT encoding. We applied the compression algorithm to two different hyper-spectral data cubes scenarios: 1. Urban scenery that has many details and thus has many oscillations in it (see section 4.2.1). Therefore, it is more difficult to compress. 2. Rural scenery, which has less details and has smoother structure, thus, "easier" to compress (see section 4.2.2). Each pixel in the source hyper-spectral cube has 16 bits.

4.2.1 Example 1: Urban scene

Figure 4.11 displays one of the $X-Z$ multipixel planes from the urban ground scene.

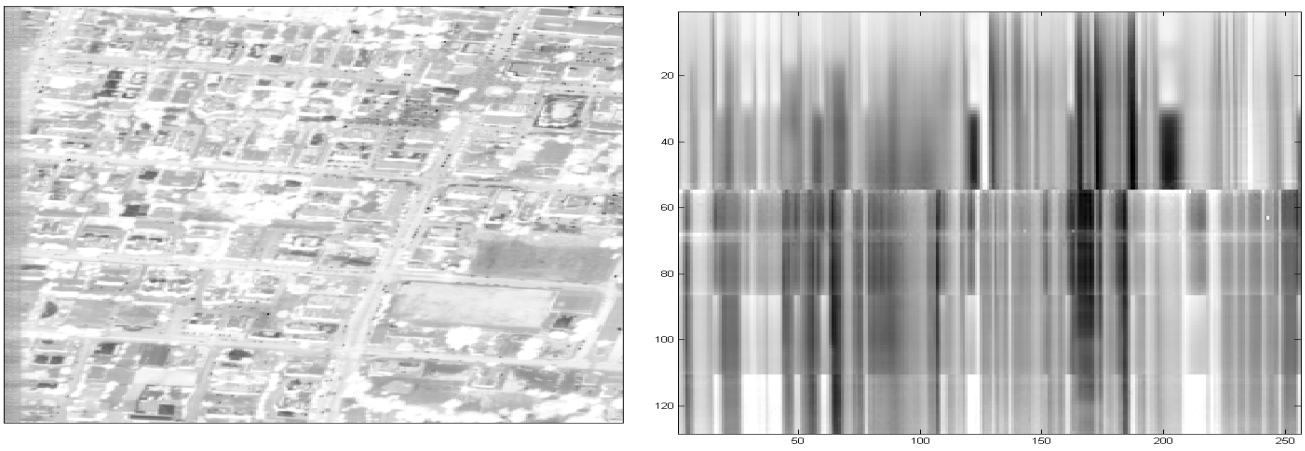

Figure 4.11: An hyper-spectral image. Left: One plane from the data cube. Right: A multipixel plane.

Table 4.2.1 compares between the performance of the hybrid scheme and the standard 2D wavelet transform followed by the application of SPIHT. The performance is given in PSNR values. They are averaged over 200 $X-Z$ planes.

bit/pixel (compression ratio)	W9/7	WButt/4	WButt/10	H9/7/16	HButt/4/16
1/4 (64)	27.47	27.52	27.52	31.06	**31.31**
1/2 (32)	31.09	31.10	31.05	36.56	**36.82**
1 (16)	36.29	36.29	35.96	43.28	**43.38**
2 (8)	44.99	44.90	44.30	51.02	**51.23**

Table 4.3: Averaged PSNR values of hyper-spectral compression of $X-Z$ planes from an urban scene.

To further improve the scheme by utilizing the Y-axis redundancy, the differences between two consecutive planes along the Y-axis were also compressed, i.e we subtracted each frame from its previous one and compressed the difference. Assume that an hyper-spectral data cube contains z wavebands where each waveband is an image of size $x \cdot y$. Then, we look at this data cube as y 2D images of sizes $z \cdot x$. The hyper-spectral camera captures y 2D images of sizes $z \times x$. Each 2D image of the y images is compressed by the application of the hybrid algorithm. The compression is done in real-time with no buffering since it is applied to each 2D image as it captured by the hyper-spectral camera. Denote each of the 2D images by I_i, $i = 1, \ldots, y$. The quality of the compression is enhanced by compressing the differences among neighboring 2D images (as in video compression) using the following procedure: I_i, $i = 1$, is compressed by the application of the hybrid transform followed by the application of SPIHT. Then, I_i,$i = 1$, is reconstructed and the difference $I_{i+1} - I_i$,$i = 1$, is

compressed. The same procedure repeats itself for $i = 3, \ldots, y$.

Table 4.2.1 exemplifies the strength of the differential compression scheme and its benefit to our hybrid transformation.

bit/pixel (compression ratio)	W9/7	WButt/4	WButt10	H9/7/16	HButt/4/16
1/4 (64)	26.72	26.84	26.76	31.80	**32.25**
1/2 (32)	31.11	31.06	30.93	37.43	**38.02**
1	(16) 37.91	37.90	37.49	43.90	**44.08**
2 (8)	46.76	46.71	46.12	51.82	**51.92**

Table 4.4: Averaged PSNR values of differential hyper-spectral compression of $X - Z$ urban plane.

Figure 4.12 displays a multipixel from the plane #200, which was reconstructed after the application of **W9/7** (left figure) and the **HButt/4/16** (right figure) transforms followed by SPIHT encoding/decoding for 0.5bpp compression ratio.

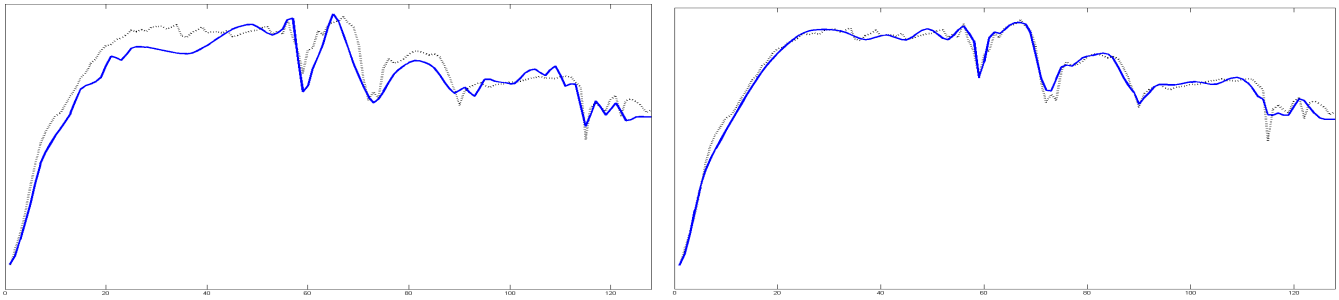

Figure 4.12: One multipixel from the multipixel plane $X - Z$ #200 is shown. Dotted line: Original. Solid line: Restored after the application of – Left: **W9/7** transform. Right: **HButt/4/16**. SPIHT encoding/decoding were used with 0.5bpp compression ratio

Almost all the "events" in the original data are present also in the reconstructed signal after the application of the hybrid transform. This is not the case when the 2D wavelet transform **W9/7** is applied.

4.2.2 Example 2: Rural scene

Figure 4.13 displays a rural ground scene and one of the $X - Z$ multipixel planes.

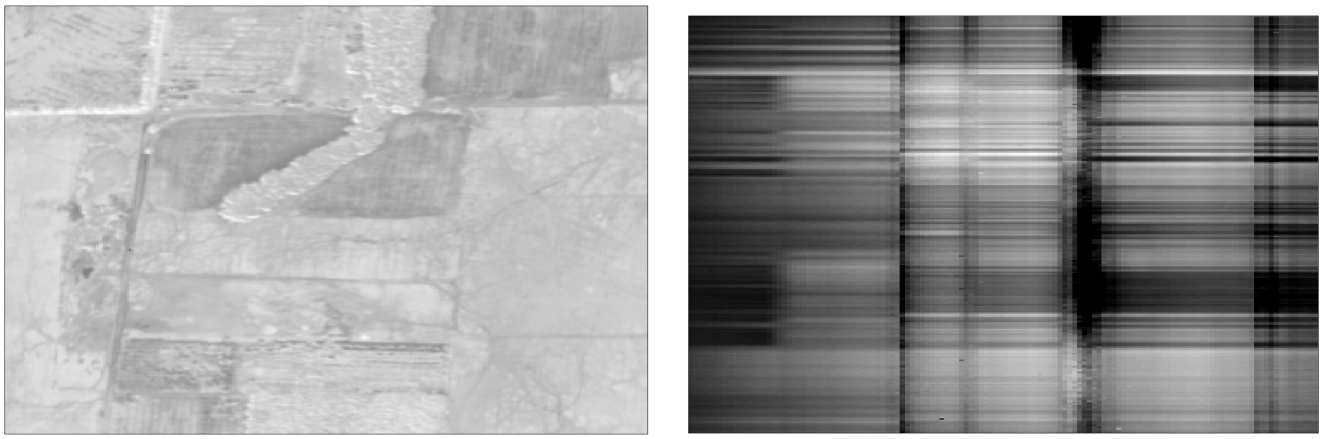

Figure 4.13: An hyper-spectral image. Left: Rural ground scene, right: X/Z multipixel plane #100.

Table 4.5 displays the PSNR values averaged over 200 $X - Z$ planes after the application of different transforms without utilizing the differences among consecutive wavebands. Table 4.6 displays the PSNR values averaged over 200 $X - Z$ planes after the application of different transforms after utilization of the the differences among consecutive wavebands.

bit/pixel (compression ratio)	W9/7	WButt/4	WButt/10	H9/7/16	HButt/4/16
1/4 (64)	29.12	28.95	28.73	31.74	**31.88**
1/2 (32)	32.43	32.30	31.91	36.33	**36.70**
1 (16)	36.94	36.81	36.46	42.06	**41.90**
2 (8)	44.02	43.81	43.24	49.75	**49.78**

Table 4.5: Averaged PSNR values of hyper-spectral compression of $X - Z$ planes of rural scene

bit/pixel (compression ratio)	W9/7	WButt4	WButt10	H9/7/16	HButt4/16
1/4 (64)	30.87	30.82	30.75	34.39	**34.58**
1/2 (32)	34.37	34.26	34.19	38.30	**38.49**
1 (16)	39.63	39.40	39.21	43.77	**43.88**
2 (8)	47.53	47.32	46.90	51.05	**51.02**

Table 4.6: Averaged PSNR values of differential hyper-spectral compression of $X - Z$ planes of rural scene.

Here, unlike the urban example, the PSNR values from the compression/decompression schemes, which utilize the difference between consecutive wavebands planes, are significantly higher than the schemes that do not utilize it. Unlike the urban scene, the rural scene is smoother and adjacent $X - Z$

planes are similar to each other.

Figure 4.14 displays a multipixel from plane #100, which were reconstructed after application of **W9/7** (left figure) and **HButt/4/16** (right figure) transforms followed by the application of SPIHT encoding/decoding with 0.5bpp.

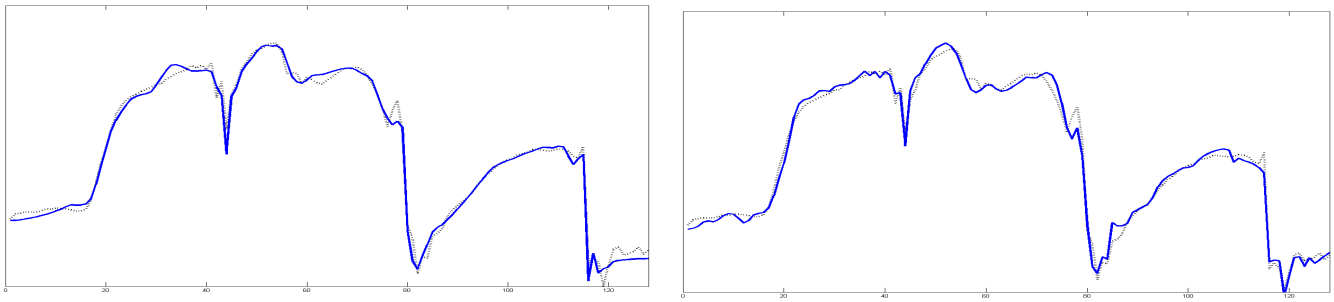

Figure 4.14: Multipixel #100 from the multipixel plane $X - Z$ is shown. Dotted line: Original. Solid line: Restored after the application of – Left: **W9/7** transform. Right: **HButt/4/16**. The transforms coefficients were encoded by SPIHT with 0.5bpp.

Conclusion: *For hyper-spectral data, the hybrid compression algorithm significantly outperforms the algorithms that are based on the 2D wavelet transforms.*

4.3 Fingerprints

The hybrid compression algorithm was applied to a fingerprint image of size 768×512, shown in Fig. 4.15, downloaded from C. Brislawn's web page http://www.c3.lanl.gov/ brislawn/index.html. Each pixel has 8 bits.

Figure 4.15: The original fingerprint image. Left: The whole image. Right: A fragment.

It has an oscillating structure in each direction. Therefore, the 2D LCT transform **LDCT/24/16** was applied followed by the SPIHT–adapted reordering of the transform coefficients. We compared between the performance of this transform and the performance of the 2D wavelet transforms and the hybrid transforms, where $Q = 24$. The achieved PSNR values are given in Table 4.7.

bit/pixel (compression ratio)	W9/7	WButt/10	H9/7/24	HButt/10/24	LDCT/24/16
1/8 (64)	23.34	24.09	23.86	**24.29**	23.41
1/4 (32)	25.71	26.43	26.44	**26.64**	26.23
1/2 (16)	29.31	**29.87**	29.90	29.83	**29.87**
1 (8)	33.29	33.77	33.80	34.08	**34.42**

Table 4.7: PSNR values for fingerprint compression.

We display in Fig. 4.16 the fragments of the reconstructed fingerprint image after the application of the 2D **W9/7** transform and the Hybrid transform **HButt/10/24** were applied and the coefficients were coded by SPIHT with 1/8 bit pet pixel. The fragment from the **HButt/10/24** transformed section retains much better the structure of the data than the fragment from the **W9/7** transform.

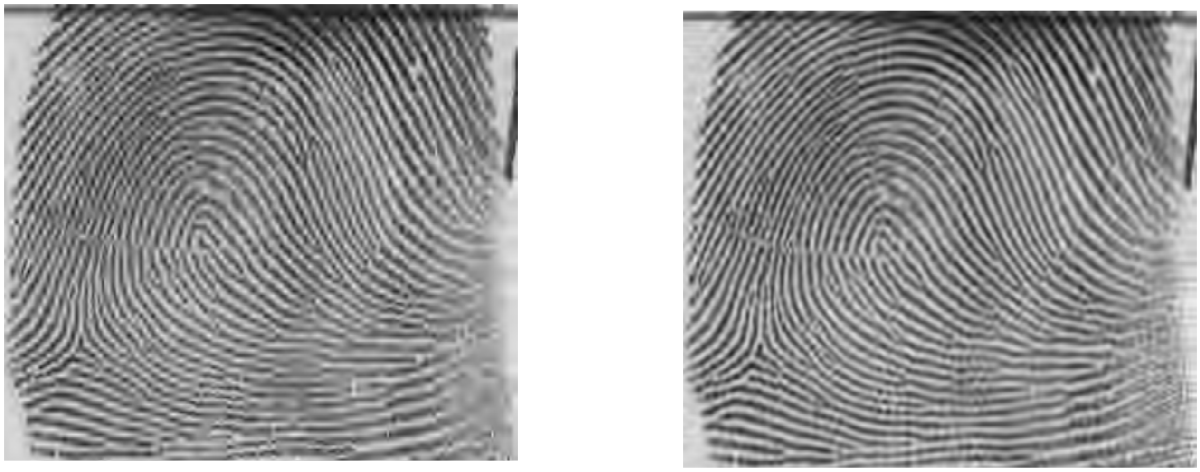

Figure 4.16: A fragment from the reconstructed fingerprint image. Left: **W9/7** transform was applied. Right: The Hybrid transform **HButt/10/24** was applied. The transforms coefficients were encoded/decoded by SPIHT with 1/8 bit per pixel compression rate

Figure 4.17 displays a fragment of column #300 from the original section versus a fragment from the reconstructed section after the application of **W9/7** (left) and **HButt/10/24** (right) transforms

followed by SPIHT encoding/decoding with 1/8 bit pet pixel.

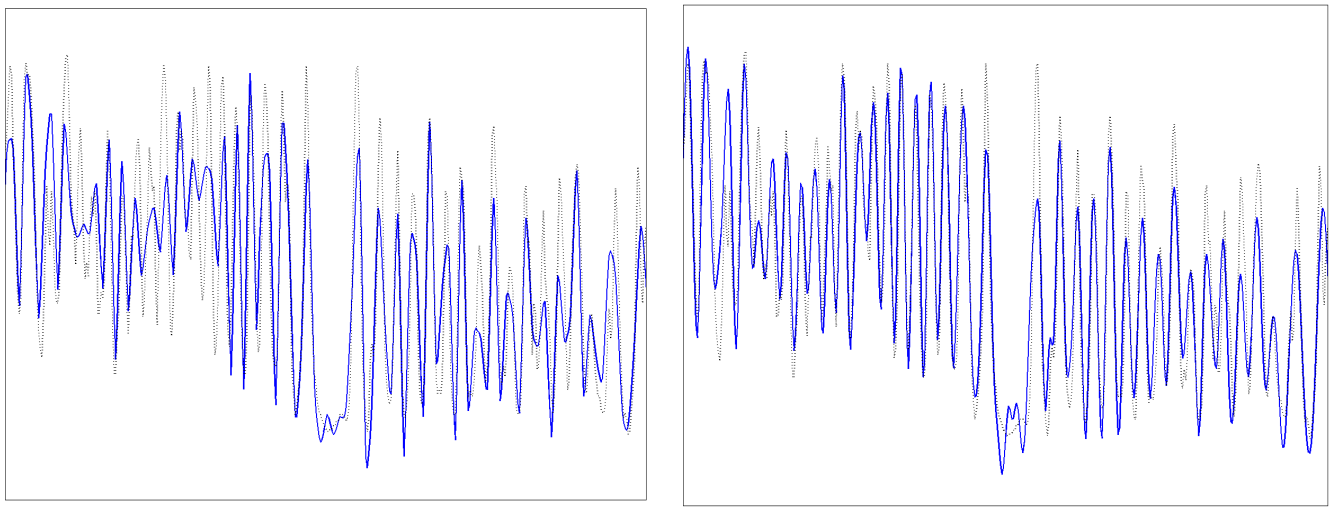

Figure 4.17: A fragment from column #300 in the fingerprint image. Dotted line: Original. Solid line: Restored after the application of **W9/7** (left) and **HButt/10/24** (right) transforms followed by SPIHT encoding/decoding with 1/8 bit per pixel.

Conclusion: For fingerprint images, the Hybrid compression algorithm outperforms the algorithm that are based on 2D wavelet transforms. The algorithm, which is based on the 2D lapped DCT, is efficient for high bitrates compression.

4.4 Compression of multimedia images

The hybrid compression algorithm is also efficient for multimedia images. Its superior performance is more evident for images that have oscillating texture. The algorithm restores the texture even at a very low bitrate. On the other hand, it sometimes produces artifacts on the boundaries between smooth and texture areas. Each pixel in these images has 8 bits.

"Barbara" image of size 512×512 and its fragment of size 300×300 are displayed in Fig. 4.18.


Figure 4.18: The original "Barbara" image . Left: The whole image. Right: A fragment.

This image comprises areas with oscillating textures. We compared between the performance of the 2D wavelet transforms and the hybrid transforms, which used an image partition into $Q = 16$ horizontal rectangles each of height 32. The achieved PSNR values are presented in Table 4.8.

bit/pixel (compression ratio)	W9/7	WButt/4	WButt/10	H9/7/16	HButt/10/16
1/8 (64)	24.69	24.42	24.65	24.97	**25.45**
1/4 (32)	27.36	27.04	27.80	27.92	**28.52**
1/2 (16)	31.08	31.17	31.83	32.07	**32.64**
1 (8)	36.33	36.52	37.41	37.39	**37.79**

Table 4.8: PSNR values from the compression/decompression of "Barbara".

We display in Figs. 4.19 and 4.20 the whole "Barbara" image and its fragment, respectively, that were reconstructed after the 2D **W9/7** transform and the Hybrid transform **HButt/10/16** were applied. The coefficients were coded by SPIHT using 1/8 bit per pixel compression rate. The reconstruction from the **HButt/10/16** transformed image retains much better the texture of the data than the **W9/7** transformed image. However some artifacts are seen on the chin of Barbara.

Figure 4.19: The reconstructed "Barbara" image. Left: **W9/7** transform was applied. Right: The Hybrid transform **HButt/10/16** was applied. The transforms coefficients were encoded/decoded by the application of the SPIHT using 1/8 bit per pixel compression rate

Figure 4.20: The fragment of the reconstructed "Barbara" image. Left: **W9/7** transform was applied. Right: The Hybrid transform **HButt/10/16** was applied. The transforms coefficients were encoded/decoded by SPIHT with 1/8 bit per pixel compression rate

Conclusions

The experimental results supports strongly our assumption that the hybrid wavelet– LCT – SPIHT algorithm with the new reordering of the transform coefficients is a new powerful tool to compress seismic data and hyper-spectral images that comprise oscillating structures. Although all the three components of the algorithm are well known, their combined operation via the reordering of the

transform coefficients outperform well known common tools. Additional flexibility of the method stems from the availability of the library of Butterworth wavelet transforms of different orders. In almost all the experiments, the hybrid transforms with the Butterworth wavelets outperform the transforms that used the popular 7/9 wavelets. Presumably, the Butterworth wavelet coefficients are better compatible with the LCT coefficients compared to the wavelets. The extension of the algorithm to compress 3D seismic or hyper-spectral cubes is straightforward. In the horizontal planes, the 2D wavelet transform is applied, while the LCT is applied to vertical directions followed by reordering the coefficients. The joint array of coefficients is the input to SPIHT encoding.

Appendix: Butterworth wavelet transforms

The construction and implementation of the Butterworth wavelet transforms were introduced in [2]. In [3] they are embedded into a general spline- based wavelets and frames framework. Here, we briefly outline the subject.

We call the sequences $\mathbf{x} \stackrel{\Delta}{=} \{x(n)\},\ n \in \mathbb{Z}$, which belong to the space l_1, (and, consequently, to l_2) discrete-time signals. The z-transform of a signal $\mathbf{x}$ is defined as $X(z) \stackrel{\Delta}{=} \sum_{n\in\mathbb{Z}} z^{-n}\, x(n)$. We assume that $z = e^{j\omega}$.

The input $x(n)$ and the output $y(n)$ of a linear discrete time shift-invariant system are linked by the discrete convolution $y(n) = \sum_{l\in\mathbb{Z}} h(n-l)x(l)$. This processing of the signal $\mathbf{x}$ is called digital filtering and the sequence $\{h(n)\}$ is called the impulse response of the filter $\mathbf{h}$. Its z-transform $H(z) \stackrel{\Delta}{=} \sum_{n\in\mathbb{Z}} z^{-n}h(n)$ is called the transfer function of the filter. Usually, a filter is designated by its transfer function $H(z)$. In the z-domain filtering is reduced to the multiplication by the transfer function: $Y(z) = H(z)X(z)$. The function $\widehat{H}(\omega) = H(e^{j\omega n})$ is called the frequency response of the digital filter.

A useful tool for the design and implementation of the biorthogonal wavelet transforms is provided by the so-called lifting scheme introduced by Sweldens [26].

Lifting scheme: decomposition

Generally, the lifting mode of the wavelet transform consists of of three steps: 1. Split. 2. Predict. 3. Update or lifting.

Split: The signal $\mathbf{x} = \{x(l)\}_{l\in\mathbb{Z}}$ is split into its polyphase components:
$\mathbf{x}_r = \{x_r(l) \stackrel{\Delta}{=} x(2l+r\}_{l\in\mathbb{Z}},\ r = 0, 1.$

Predict: The even polyphase component is filtered by some low-pass prediction filter $\tilde{F}(1/z)$, in order for the filtered version of $\mathbf{x}_0$ to predict the odd component $\mathbf{x}_1$. Then, the existing array $\mathbf{x}_1$ is replaced by the array $\mathbf{a^1}$, which is the difference between $\mathbf{x}_1$ and the predicted array.

In the z-domain, the operations are described as $\tilde{A}^1(z) = X_1(z) - \tilde{F}(1/z)X_0(z)$.

Update (lifting): The new odd array is filtered by a low-pass *update* filter, which we prefer to denote $F(z)/2$. The filtered array is used to increase the smoothness of the even array $\mathbf{x}_0$:

$$\tilde{Y}^0(z) = X_0(z) + \frac{1}{2}F(z)\tilde{Y}^1(z).$$

Provided that the filter F is properly chosen, the even array $\mathbf{x}_0$ is transformed into $\tilde{\mathbf{y}}_0$, which is a a smoothed and downsampled replica of $\mathbf{x}$.

Finally, the smoothed array $\tilde{\mathbf{y}}^0$ and the array of details $\tilde{\mathbf{y}}^1$ are obtained.

Lifting scheme: reconstruction

One of the most attractive features of lifting schemes is that the reconstruction of the signal $\mathbf{x}$ from the arrays $\tilde{\mathbf{y}}^{\mathbf{0}}$ and $\tilde{\mathbf{y}}^{\mathbf{1}}$ is implemented by reverse decomposition:

Undo Lifting - The even polyphase component $X_0(z) = \tilde{Y}^0(z) - \frac{1}{2}F(z)\tilde{Y}^1(z)$ is restored.

Undo Predict - The odd polyphase component

$$X_1(z) = \tilde{Y}^1(z) + \tilde{F}(1/z)X_0(z).$$

is restored.

Undo Split - The last step is the standard restoration of the signal from its even and odd components. In the z-domain, it appears as $X(z) = X_1(z^2) + z^{-1}X_0(z^2)$.

The predict and update filters

Selecting the predict and update filters $\tilde{F}(z)$ and $\tilde{F}(z)$, we can design a variety of wavelet transforms. We propose to use the following parameterized family of the filters, which originates from the theory of discrete splines. Denote $\rho(z) \stackrel{\Delta}{=} z + 2 + 1/z$. Then, the predict filter of order $2m$ we define as follows

$$\tilde{F}_m(z^2) = z\frac{\rho^m(z) - \rho^m(-z)}{\rho^m(z) + \rho^m(-z)}.$$

The filters $\tilde{F}_m(z^2)$ are closely related to Butterworth filters [16], which are widely used in signal processing.

Usually for the update we take the same filter $F(z^2) = \tilde{F}_m(z^2)$. The wavelets which arise from these transforms have $2m$ vanishing moments that is their inner products with the polynomials of degree $2m-1$ are zero.

The above filters are IIR but their impulse response decays exponentially. They are implemented by the fast recursive algorithm [2]. Herewith the computational cost of the implementation do not exceed the cost of the implementation of FIR filters with similar properties.

References

[1] A. Z. Averbuch, F. Meyer, J-O. Stromberg, R.Coifman, A. Vassiliou, "Efficient Compression for Seismic Data", IEEE Trans. on Image Processing, vol. 10, no. 12, pp. 1801-1814, 2001.

[2] A. Z. Averbuch, A. B. Pevnyi and V. A. Zheludev *Butterworth wavelet transforms derived from discrete interpolatory splines: Recursive implementation*, Signal Processing, **81**, (2001), 2363-2382.

[3] A. Averbuch, V. Zheludev, *Wavelet and frame transforms originated from continuous and discrete splines*, in "Advances in Signal Transforms: Theory and Applications",J. Astola and L. Yaroslavsky eds. , pp. 1-56, Hindawi Publishing Corp., NY, 2007.

[4] A. Averbuch, G. Aharoni, R. Coifman,M. Israeli, *Local cosine transform - A method for the reduction of blocking effects in JPEG*, Journal of Mathematical Imaging and Vision, Special Issue on Wavelets, Vol. 3, pp. 7-38, 1993.

[5] C. M. Brislawn, Classification of Nonexpansive Symmetric extension transforms for multirate filter banks *Applied and Computational Harmonic Analysis*, **3** (1996), No. 4, 337-357. 337-357.

[6] C. M. Brislawn, *The FBI Fingerprint Image Compression Specification*, in "Wavelet Image and Video Compression", ed. P. Topivala, Springer, 2002, 271-288.

[7] Tseng, Y.H., H.K. Shih, and P.H. Hsu, *Hyperspectral Image Compression Using Three-dimensional Wavelet Transformation*, Proceedings of the 21th Asia Conference on Remote Sensing, 2000, pp. 809-814.

[8] V. A. Zheludev, D. D. Kosloff, E. Y. Ragoza, Compression of segmented 3D seismic data, International Journal of Wavelets, Multiresolution and Information Processing, vol. 2, no. 3, (2004), 269-281.

[9] James E. Fowler, Justin T. Rucker, *Three-Dimensional Wavelet-Based Compression of Hyperspectral Imagery*, in "Hyperspectral Data Exploitation". Ed. Chein-I Chang, John Wiley & Sons, (2007), 379-407.

[10] A. Cohen, I. Daubechies and J.-C. Feauveau, Biorthogonal bases of compactly supported wavelets, *Commun. on Pure and Appl. Math.*' 45:485-560, 1992.

[11] X. Tang and W. A. Pearlman, *Three-dimensional wavelet–based compression of hyperspectral images*, in "Hyperspectral data compression", Eds. G. Motta, F. Rizzo, and J. A. Storer, Kluwer Acad. Publ., 2006, 273-308.

[12] G. Motta, F. Rizzo, and J. A. Storer eds "Hyperspectral data compression", Kluwer Acad. Publ., 2006, 273-308.

[13] R. R. Coifman and Y. Meyer, "Remarques sur l'analyse de Fourier a fenêtre", C. R. Acad. Sci., pp. 259-261, 1991.

[14] P. L. Donoho, R. A. Ergas, and J. D. Villasenor, High-performance seismic trace compression, In 65th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, (1995), 160-163.

[15] I. Daubechies, 1992,Ten lectures on wavelets. SIAM.

[16] A. V. Oppenheim and R. W. Schafer. *Discrete-time signal processing.* Englewood Cliffs, New York, Prentice Hall, 1989.

[17] E. Feig and S. Winograd, "Fast algorithms for the discrete cosine transform", IEEE Trans. Sign. Proc., vol. 40, pp. 2174-2193, 1992.

[18] JPEG 2000 Standard, http://www.jpeg.org/jpeg2000/.

[19] M. F. Khéne, S. H. Abdul-Jauwad, "Efficient seismic compression using the lifting scheme", In 70th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, (2000), 2052-2054.

[20] F. G. Meyer, *Fast compression of seismic data with local trigonometric bases,* Proc. SPIE **3813**, Wavelet Applications in Signal and Image Processing VII, (A. Aldroubi, A. F. Laine; M. A. Unser; Eds.) (1999), 648-658.

[21] F. G. Meyer, *Image Compression With Adaptive Local Cosines: A Comparative Study*, IEEE Trans. on Image Proc., vol. 11, no. 6, June 2002, 616–629.

[22] A. Said and W. W. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees" IEEE Trans. on Circ. and Syst. for Video Tech., vol. 6, pp. 243-250, 1996.

[23] S. Mallat *A wavelet tour on signal processing,* Acad. Press, 1999.

[24] G. Matviyenko, *Optimized local trigonometric bases*, Applied and Computational Harmonic Analysis, 3, 301-323, 1996.

[25] J. M. Shapiro, "Embedded image coding using zerotree of wavelet coefficients", IEEE Trans. Sign. Proc., vol. 41, pp. 3445-3462, 1993.

[26] W. Sweldends *The lifting scheme: A custom design construction of biorthogonal wavelets*, Appl. Comput. Harm. Anal. **3(2)**, (1996), 186-200.

[27] A. Vassiliou, and M. V. Wickerhauser, "Comparison of wavelet image coding schemes for seismic data compression", In 67th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, (1997),1334-1337.

[28] Y. Wang, R.-S. Wu, "Seismic data compression by an adaptive local cosine/sine transform and its effect on migration", Geophysical Prospecting, vol. 48, pp. 1009-1031, 2000.

[29] Z. Xiong, O. Guleryuz and M. T. Orchard, "A DCT-based embedded image coder", IEEE Signal Processing Letters., vol. 3, pp. 289-290, Nov. 1996.

[30] Z. Xiong, K. Ramchadran, M. T. Orchard, and Y.-Q. Zhang, "A comparative study of DCT- and wavelet-based image coding", IEEE Trans. on Circ. and Syst. for Video Tech., vol. 9, pp. 692-695, 1999.

[31] V. A. Zheludev, D. Kosloff, E. Ragoza "Fast Kirchhoff migration in wavelet domain", Exploration Geophysics, vol. 33, pp. 23-27, 2002.

[32] B-J. Kim, Z. Xiong, and W. A. Pearlman, "Low Bit-Rate Scalable Video Coding with 3D Set Partitioning in Hierarchical Trees (3D SPIHT)," IEEE Trans. Circuits and Systems for Video Technology, Vol. 10, pp. 1374-1387, Dec. 2000.